SENSOR VALIDATION

T.W. Bickmore
Aerojet Propulsion Systems
P.O. Box 13222
Sacramento, CA 35813

PRA-SA-APP-NASA/LEWIS 21 APRIL 1994

Abstract

The NASA Lewis Research Center and Aerojet Propulsion Division have been working together to develop software capable of detecting sensor failures on liquid rocket engines in real time, and with a high degree of confidence. This software could be used in a rocket engine controller to prevent the erroneous shutdown of an engine due to sensor failures which would otherwise be interpreted as engine failures by the control software.

The approach taken combines analytic redundancy with Bayesian belief networks to provide a solution which has well-defined real-time characteristics, well-defined error rates, and is scalable to validate any number of engine sensors. Analytical redundancy is a technique in which a sensor's value is predicted by using values from other, usually non-redundant, sensors and known or empirically derived mathematical relations. A set of sensors and a set relationships among them form a network of cross-checks which can be used to periodically validate all of the sensors in the network. Bayesian belief networks provide a mathematically sound method of determining if each of the sensors in the network is valid, given the results of all of these cross-checks.

This approach has been codified in an algorithm which has been successfully demonstrated on a rocket engine controller in real-time on the Technology Test Bed at the NASA Marshall Space Flight Center. Current efforts are focused on extending the demonstration system to provide a real-time validation capability for approximately 100 sensors on the Space Shuttle Main Engine.

I. Introduction

The safety and reliability of rocket engines would be enhanced if engine controllers and advanced safety systems could determine if sensors were supplying faulty data. This ability, termed sensor data validation, could prevent the controller or safety system from making critical decisions, such as the decision to shut an engine down, on the basis of data from anomalous or failed sensors.

Efforts to develop an approach to real-time sensor data validation (SDV) for liquid rocket engines have evolved over four years, from conceptual design (1990) to software implementation and test in a rocket engine controller on the Technology Test Bed test stand (1992). More recent efforts have focused on scaling up the capability demonstrated on Technology Test Bed to validate the majority of sensors used for redlines, control and advanced anomaly detection on the Space Shuttle Main Engine (SSME).

The remainder of this paper is organized as follows: Section II describes the initial study performed on this program which led to the development of the current approach; Sections III and IV present the theoretical background and implementation details of the approach; Section V presents the experimental results obtained to date; and Sections VI and VII present on-going and planned work on the program.

II. System Architecture Study

In 1990 a System Architecture Study of SDV was performed by Aerojet which reviewed common sensor failure modes on the SSME, the data validation process used by SSME data analysts at MSFC, and a number of alternative approaches to automating SDV for post-test/post-flight data analysis [1].

The approaches to SDV reviewed included range and rate limit checking, various pattern-matching techniques, and analytical redundancy. The conclusion of this study was that no single algorithmic method should be used for SDV; rather several methods should be used to analyze sensor data and the results integrated or "fused" into a final conclusion regarding the integrity of each sensor. Several approaches to information fusion were also reviewed for their applicability to SDV, including binary logic, ad-hoc certainty factors, Dempster-Shafer theory, and Bayesian belief networks [2]. Bayesian belief networks were selected as the best strategy, since they were believed to be the most mathematically sound approach to information fusion.

III. Probabilistic Approach to Analytical Redundancy

Real-time sensor data validation was targeted as a demonstration application for Aerojet's Advanced Rocket

Engine Controller (AREC), developed on Aerojet's Integrated Controls and Health Management IR&D in 1991. The approach taken combined analytical redundancy with Bayesian information fusion techniques to achieve a solution which has well-understood false alarm and missed detection error rates, operates within hard time constraints, and is scalable to validate any number of sensors.

Analytical redundancy is a technique in which a sensor's value is predicted by using values from other, usually non-redundant, sensors and known or empirically derived relations among the sensor values. For example, Fig. 1 shows a relation among three sensor values using a standard formula for fluid line resistance. Relations can also be empirically derived using standard statistical regression techniques. The simplest form of these empirical relations is a linear equation relating two sensor values, as shown in Fig. 2. In general, a relation is used to provide validation information for all related sensors.

A group of sensors and a set of relations among them define a network. Fig. 3 shows a very simple example of a sensor validation network for three parameters on the SSME.

The difference between a value predicted using a relation and a directly sensed value is called a *residual,* and is a measure of the quality of the relation, given that the sensors involved are known to be working properly. In the approach taken in this work, one or more relations are defined for every sensor in the network which relate its value to the values of one or more other sensors in the network. The mean and standard deviation of the relation residuals (evaluated on normal engine test firing data) are also computed.

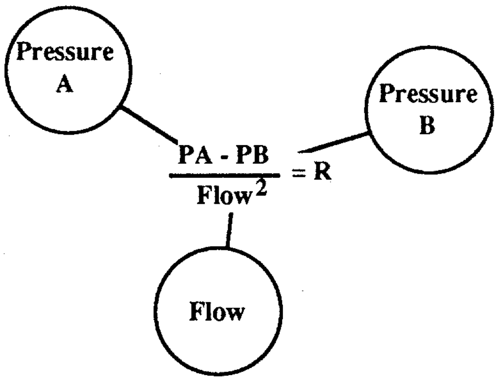

Fig. 1. Example Characteristic Relation

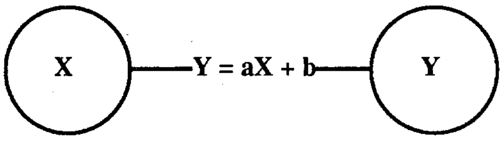

Fig. 2. Example Statistical Relation

Given this information, a validation algorithm could sample sensor values every controller cycle during an engine firing and determine if each of the relations holds or not by thresholding on a particular residual, such as three standard deviations. Once the status of every relation in the network has been determined to either "hold" or "not hold", the validation algorithm makes a conclusion about the validity of each sensor in the

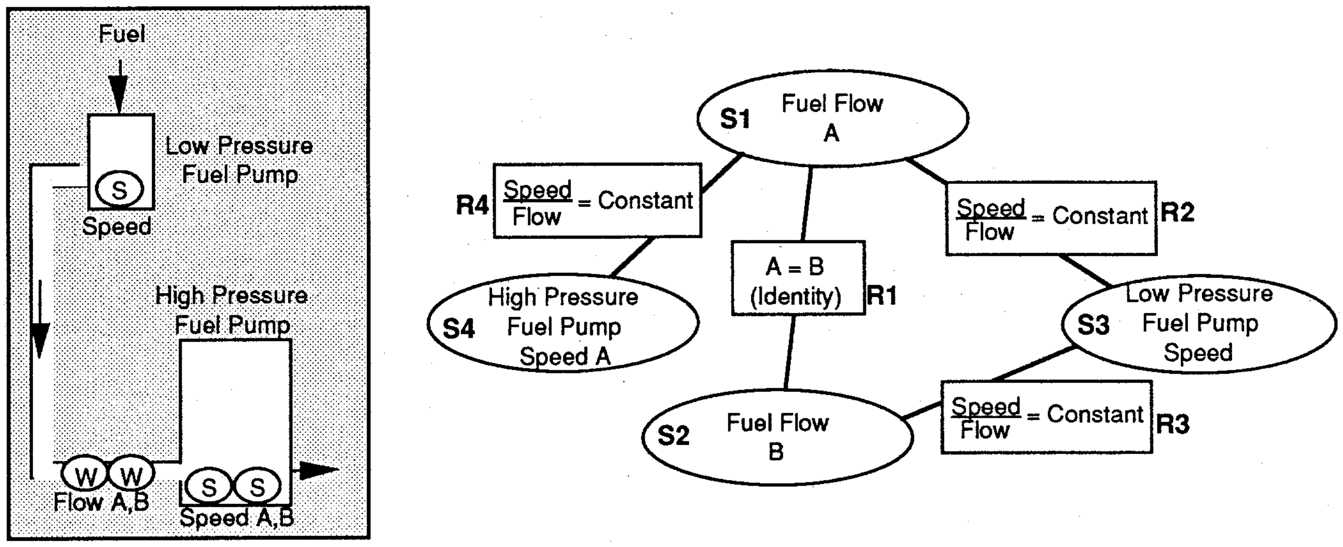

Fig. 3. Example Sensor Validation Network

network (the one-cycle decision problem). Conclusions made during several consecutive controller cycles may be fused together in order to disqualify a sensor (the multi-cycle decision problem). Fig. 4 summarizes this overall approach.

Before this general approach could be implemented, several questions needed to be answered:

- How many relations are needed to validate a sensor?
- How many of a sensor's relations need to hold in order to validate the sensor during one controller cycle?
- What threshold should be used on the individual relation residual tests?
- Should sensor value averaging or other multi-cycle strategies be used?
- Can a scalable approach to validation be developed which will work with any number of sensors?
- Do all relations need to be evaluated every cycle to validate all sensors?

III. Bayesian Analysis

Bayesian probability theory provides a formal framework within which the questions posed above can be answered. Bayesian probability theory provides a mathematically sound approach to the problem of *information fusion* — the combination of evidence from several sources into a single, consistent model. In information fusion, uncertainties in the sources of evidence (i.e., inaccuracies in the sensors or uncertainties in the fault detection algorithms themselves) are explicitly modeled and accounted for.

A Bayesian Belief Network is a graphical representation of a joint probability distribution of a set of random variables [2,3]. As an example, the validation network shown in Fig. 3 can be represented as the Belief Network shown in Fig. 5. In this network, the nodes S1, S2, S3, and S4 represent the status of the respective sensors (i.e. whether they are working or not), while the nodes R1, R2, R3, and R4 represent whether an analytical redundancy relationship currently holds between the sensors or not. Connections in the network represent influences between variables. In Fig. 3, for example, a failure in sensor S1 would influence the expected probability distributions on the status of relations R1, R2, and R4.

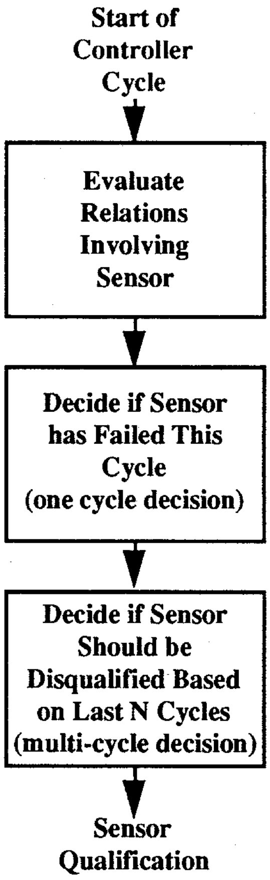
Start of Controller Cycle
Evaluate Relations Involving Sensor
Decide if Sensor has Failed This Cycle (one cycle decision)
Decide if Sensor Should be Disqualified Based on Last N Cycles (multi-cycle decision)
Sensor Qualification

Fig. 4. Overall Approach to Real-Time Sensor Data Validation

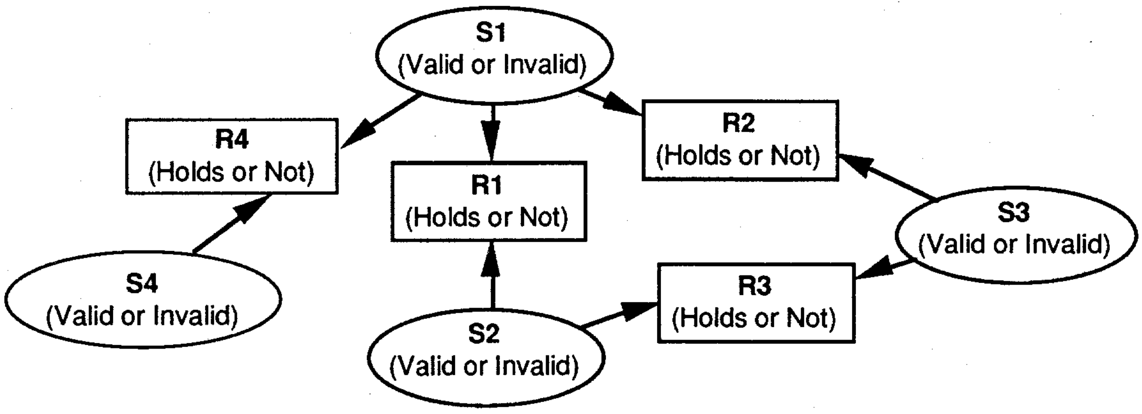

Fig. 5. Example Sensor Validation Network Cast into a Bayesian Belief Network for Analysis

Given the Belief Network shown in Fig. 5, the probability of each sensor being valid given the current status of all relations can be derived. These equations can then be used to answer the questions posed above, and to develop a mathematically sound approach to sensor data validation.

The following assumptions were made in the 1991 activity in order to conduct the Bayesian analysis:

- Although several sensors may fail during a firing, two sensors cannot fail during a single controller cycle. The likelihood of two or more sensors initiating a failure at the same instant in time is very remote, and it would greatly complicate the validation system to accommodate such simultaneous multiple-point failures.
- Once a sensor is determined to have failed, it will stay failed and will not be used again in any future calculations.
- The probability of a sensor failing can be determined from its Mean Time Between Failure (MTBF) as follows.

  $$P(\text{Sensor Invalid})^{*} = \text{CycleTime} / \text{MTBF}.$$

  This measure is assumed to be constant for the duration of a single engine firing.
- When a sensor fails it emits random values. This is a very conservative assumption, and is a more difficult failure mode to detect than a hard failure (i.e., if the algorithm is able detect the random failure mode with a high degree of confidence, it will also be able to detect hard and drift failures). This is an admission that a failed sensor has some small probability of emitting a value which is within the realm of "reasonableness" for the parameter being measured.

The conditional probabilities required to fully define the Belief Network shown in Fig. 5 were derived from the assumptions given above. The following derivations assume that all relations are binary (i.e., integrate information from two sensors).

The probability of a relation holding, given that all of the related sensors are working, is determined by the threshold level placed on the relation. Thus,

$$P(\text{RelationHolds} \mid \text{Sensor}_1\text{Valid}, \text{Sensor}_2\text{Valid}) = K$$

where K is a quantile of the normal distribution (e.g., for a 3 standard deviation relation threshold, K = 0.997).

Since a failed sensor emits random values, there is still some probability, P_n, that a given reading may fall within the normal range of values for the sensor, causing the relation to continue to hold. (Note that for hard sensor failures $P_n = 0$.) If this normal range is taken to be 3 standard deviations, then

$$P_n = \frac{2 \text{ x } 3 \text{ x Standard Deviation}}{\text{Range of Sensor}}$$

For SSME sensors, Pn has been empirically determined to be have an average value of 0.22 (although it is slightly different for each sensor). The probability of a binary relation holding given that one of its sensors has failed is thus

$$P(\text{RelationHolds} \mid \text{Sensor}_1\text{Invalid},\text{Sensor}_2\text{Valid}) = P_n$$

$$P(\text{RelationHolds} \mid \text{Sensor}_1\text{Valid}, \text{Sensor}_2\text{Invalid}) = P_n$$

Similarly, the probability of a binary relation holding given that both of its sensors have failed is

$$P(\text{RelationHolds} \mid \text{Sensor}_1\text{Invalid}, \text{Sensor}_2\text{Invalid}) = P_n^2$$

The probabilities given above yield the following joint probability distribution for the network shown in Fig. 5.

$$\begin{gathered} P(S1,S2,S3,S4,R1,R2,R3,R4) = \\ P(S1) \text{ x } P(S2) \text{ x } P(S3) \text{ x } P(S4) \text{ x} \\ P(R1|S1,S2) \text{ x } P(R2|S1,S3) \text{ x } P(R3|S2,S3) \text{ x} \\ P(R4|S1,S4) \end{gathered}$$

Given the joint distribution, the goal is to determine the probability of any one sensor working given the status of all relations in the network (this is the basis for the real-time, one-cycle decision problem). This can be achieved by using Bayes' rule. For example, after measurements for S1, S2, S3, and S4 have been taken, and relations R1, R2, R3, and R4 have been evaluated to determine whether they hold or not, the probability of sensor S1 working can be determined as follows.

$$P(S1|R1,R2,R3,R4) = \frac{P(S1,R1,R2,R3,R4)}{P(R1,R2,R3,R4)}$$

Given the ability to compute the probability of a sensor being valid or not given the status of all relations in the network (as in the above equation), an optimum one-cycle decision strategy can be developed by simply thresholding on this probability. For example, Table 1 shows the validation probabilities for sensor S1 given that the

* "P(A)" can be read as "the probability of A being true". "P(A,B)" can be read as the "the probability of A and B being true." "P(A | B)" can be read as "the probability of A being true given that B is true". "P(A | B,C)" can be read as the "the probability of A being true given that B and C are true".

MTBF of S1, S2, S3, and S4 in Fig. 5 is 30 minutes, the relation residual threshold for R1, R2, R3, and R4 is 3 standard deviations, and P_n is 0.22. From this table it can be seen that the optimum strategy, given these assumptions, is to disqualify sensor S1 when relations R1, R2, and R4 do not hold.

There are two measures of quality for any validation algorithm: the false alarm and missed detection rates (equivalent to Type I and Type II errors in statistics, respectively). The false alarm rate is the probability that the validation system will disqualify a sensor, when it is in fact working correctly. The missed detection rate is the probability that the validation system will qualify a sensor, when it has in fact failed (this is related to the notion of sensitivity). These rates can be computed for the one-cycle decision strategy described above. The false alarm rate for sensor S1 is the sum of

P(S1=Valid,S2,S3,S4,R1,R2,R3,R4)

in all situations in which the validation system decides to disqualify S1.

Similarly, the missed detection rate for sensor S1 is the sum of

P(S1=Invalid,S2,S3,S4,R1,R2,R3,R4)

in all situations in which the validation system decides to validate S1.

These two quality measures were used to evaluate many alternative answers to the questions posed above. The results indicated that:

- At least three relations involving a sensor's value are required to provide enough information to disqualify the sensor.
- The number of relations involving a sensor's value which must be violated in order to disqualify the sensor varies with the number of relations. For example, in the network shown in Fig. 5 in which sensor S1 is involved in three relations, all three relations must be found not to hold before the common sensor can be disqualified.
- A 3 standard deviation residual threshold should be used on all relations to determine if they hold or not.
- A multi-cycle decision strategy must be used in order to get the error rates below acceptable levels. The best strategy evaluated was a 3-of-5 strategy, in which a sensor must be judged bad (using the one-cycle strategy) on at least three of the last five controller cycles before it can be conclusively disqualified.

Of the results obtained, the most significant was that only the relations directly bearing on a sensor need to be evaluated in order to validate the sensor. For example, in the network shown in Fig. 5 only relations R1, R2, and R4 need to be considered when validating S1.

Given this, and the fact that a voting table can be constructed which specifies the number of those relations which must be violated before the sensor can be disqualified, an algorithm can be designed which only evaluates relations for a particular sensor until it is impossible to disqualify it. For example, when validating sensor S1 in the network shown in Fig. 3, the relations R1, R2, and R4 can be examined in sequence, but as soon as one is found to hold, the validation process for S1 can stop because it is impossible to disqualify it (i.e., all three relations must be violated in order to disqualify a sensor with three relations). Thus, all relations in the network do not need to be evaluated every cycle.

P(S1=Valid|R1=Holds,R2=Holds,R3=Holds,R4=Holds)	= 1
P(S1=Valid|R1=Holds,R2=Holds,R3=Holds,R4=NotHold)	= 0.9997204
P(S1=Valid|R1=Holds,R2=Holds,R3=NotHold,R4=Holds)	= 1
P(S1=Valid|R1=Holds,R2=Holds,R3=NotHold,R4=NotHold)	= 0.9997204
P(S1=Valid|R1=Holds,R2=NotHold,R3=Holds,R4=Holds)	= 0.9997191
P(S1=Valid|R1=Holds,R2=NotHold,R3=Holds,R4=NotHold)	= 0.7523449
P(S1=Valid|R1=Holds,R2=NotHold,R3=NotHold,R4=Holds)	= 0.9998867
P(S1=Valid|R1=Holds,R2=NotHold,R3=NotHold,R4=NotHold)	= 0.8828096
P(S1=Valid|R1=NotHold,R2=Holds,R3=Holds,R4=Holds)	= 0.9997191
P(S1=Valid|R1=NotHold,R2=Holds,R3=Holds,R4=NotHold)	= 0.7523449
P(S1=Valid|R1=NotHold,R2=Holds,R3=NotHold,R4=Holds)	= 0.9998867
P(S1=Valid|R1=NotHold,R2=Holds,R3=NotHold,R4=NotHold)	= 0.8828096
P(S1=Valid|R1=NotHold,R2=NotHold,R3=Holds,R4=Holds)	= 0.7515119
P(S1=Valid|R1=NotHold,R2=NotHold,R3=Holds,R4=NotHold)	= 0.002574868
P(S1=Valid|R1=NotHold,R2=NotHold,R3=NotHold,R4=Holds)	= 0.9227433
P(S1=Valid|R1=NotHold,R2=NotHold,R3=NotHold,R4=NotHold)	= 0.01009216

Table 1. Example Validation Probabilities for Sensor S1

The maximum number of relations which can be expected to fail per controller cycle can be computed and translated into a hard upper bound on processing time for the validation system. As an example, assume we are validating $S = 100$ sensors, each of which has 4 binary relations, with a total of $R = 200$ relations, and that the Bayesian analysis indicates that the one-cycle strategy should only flag a bad sensor when all 4 of the sensor's relations fail.

Assuming that at most one sensor can fail on a given controller cycle, the maximum number of relations which need to be evaluated each cycle is given by the following:

- Assuming that one sensor did fail on a given cycle, the number of relations that need to be evaluated to confirm the failure is simply the number of immediate relations that the sensor has (in the worst case they would all need to be checked). For a sensor with four relations, all four would need to be evaluated in order to disqualify the sensor.

- For each of the remaining valid sensors, the first relation always needs to be evaluated. However, we can compute the probability of a given number of additional relations failing and pick the smallest number that gives us the reliability we want. The probability of more than r relations out of the unevaluated $N = R - S - 3 = 97$ relations in the network failing (due to noise and modelling errors) is:

$$\sum_{i=r+1}^{N} \binom{N}{N\text{-}i} \times \text{P(RelationHolds)}^{N-i} \times \text{P(RelationNotHold)}^{i}$$

  This is a sum of binomial probabilities, which for large N and small P(RelationNotHold) can be approximated by a sum of Poisson probabilities, with μ=N x P(RelationNotHold). Given that N = 97 and P(RelationNotHold)=0.003 (for a 3-standard-deviation threshold), making μ=0.291. A table of Poisson probability sums indicates that at most nine additional relations would need to be checked to yield a very high degree of confidence.

Thus, for each of the 99 valid sensors, one relation must always be checked, and we will allow an additional nine to be checked in the overall network to guarantee a high level of confidence. For the one failed sensor, all four of its relations must be checked. Thus, in the worst case, a total of 100 + 3 + 9 = 112 relations, or 56% of all relations, need to be evaluated on any given cycle (this ratio decreases as more relations per sensor are used). The small number of relations which need to be evaluated each cycle, coupled with the fact that only the relations directly involving a sensor need to be evaluated for validation, allowed an algorithm to be developed which is entirely scalable (i.e., will work with a large number of sensors and relations).

These results are based on our assumptions about the accuracy and reliability of the sensors on the SSME. Although studies have shown that these results are insensitive to small changes in the assumed parameter values (corroborated by De Bruyne [4]), large changes would require a new analysis (e.g., if the system were to be used to validate sensors on a power plant). In particular, order-of-magnitude changes in sensor reliabilities would require a re-analysis.

IV. Software Design

The algorithm and data structures for the core sensor validation routine which performs the one-cycle decision making are outlined in Fig. 6 and Fig. 7, respectively. Every controller cycle, each sensor is checked in sequence. A sensor check consists of evaluating all of the relations which directly bear on the sensor until a conclusion about its validity can be made. Typically, this will involve evaluating a very small number of relations and then stopping when it becomes impossible to disqualify the sensor. When a sensor is permanently disqualified, all relations which use its value are deactivated. This ensures that the system will not try to perform validation using data from a failed sensor. Thus, the algorithm keeps track of which relations are active and which are inactive, and will continue to validate a sensor even when fewer relations are available. Sensors are always allocated at least one more relation than indicated by the Bayesian analysis so that a given sensor can still be validated following the disqualification of one or more of the sensors it is cross-checked against.

Several additional software modules were developed to augment the core one-cycle validation routine. These include:

- Steady-State Detection — Detects when the engine has reached one of a known set of steady-state conditions.

- Dynamic Relation Biasing — In order to get the sensitivity required to detect sensor failures before hard limits (redlines) were exceeded, the significance of engine-to-engine variations in operating conditions had to be understood and addressed. To handle this, the system took several data samples as it entered each steady-state condition and biased the relations accordingly. This biasing was limited, however, to prevent accommodating data from sensors which may have failed during transients (i.e., the bias term was itself thresholded).

```
OneCycleValidate(Sensor)
   Passed ← 0
   Validated ← False
   NumActiveRelations ←
       CountActiveRelations(Sensor.Relation_List)
   DO for each Relation in Sensor.Relation_List UNTIL Validated
     IF(Relation.Status is Active) THEN
            IF(Relation.Eval_Function()) THEN
                   Passed ←  Passed + 1
                   IF(Passed ≥ PassTable[NumActiveRelations]) THEN
                          Validated ←  True
   RETURN(Validated)
```

Fig. 6. One-Cycle Sensor Validation Algorithm

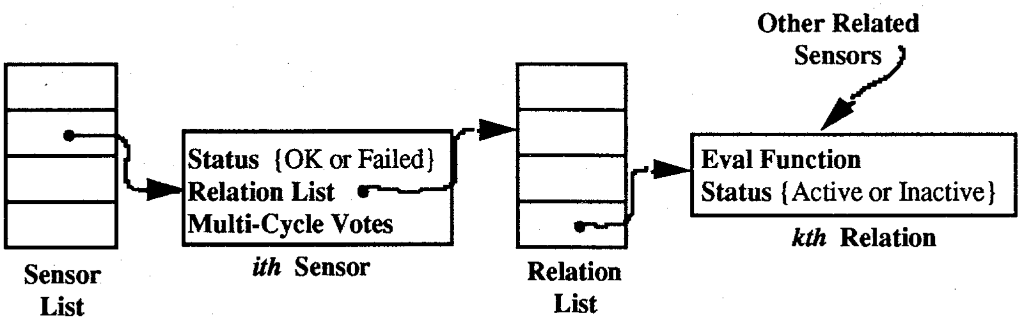

Fig. 7. Primary Data Structures Used in Sensor Validation Software

The AREC-hosted SDV system software consisted of about 1,000 lines of C code. It was developed on a Sun SPARCstation using recorded sensor data from SSME tests. A network was used to validate the High Pressure Fuel Turbine Discharge Temperature (HPFT DS T) sensors using six parameters and eleven binary empirical relations.

The models used in the AREC-hosted SDV system were empirically derived, binary models, based on data from eight nominal test firings. In general, either linear or cubic models were derived. Most relationships involving only pressures and/or speeds appeared linear when cross-plotted. However, relationships involving temperatures (particularly the High Pressure Fuel Turbine Discharge Temperatures) appeared to have a cubic relationship when cross-plotted.

Simulation Laboratory Tests

Following extensive non-real-time testing of the algorithm, the sensor validation software was integrated into the Advanced Rocket Engine Controller and set up in Aerojet's Real-Time Simulation Laboratory. This facility is based on an AD-100 multiprocessor computer, which is capable of either simulating engine firings or replaying data from engine firings in real-time. The AD-100 was programmed to replay recorded data from 10 SSME firings in order to test the AREC and the real-time characteristics of the sensor validation system. The system correctly monitored nine normal SSME tests in real time without any false alarms being generated. The system correctly detected a hard failure in HPFT DS T on a tenth SSME test dataset [5].

In order to evaluate the sensitivity of the system to "soft" sensor failures in a real-time environment, a series of tests were run in which a slow drift in HPFT DS T (high or low) was simulated by the AD-100 computer while all other sensors were held at their nominal values (engine test data was not used for these tests). The point at which the system disqualified the sensor was then recorded. These tests indicated that the system had adequate sensitivity to soft failures.

Real-Time Validation on the Technology Test Bed

Following tests in Aerojet's Real-Time Simulation Laboratory, the AREC-based sensor validation system was installed in the Technology Test Bed blockhouse at MFSC to receive and analyze real-time data from Technology Test Bed hot-fire tests. The same validation network and sensor validation software configuration used in the Simulation Laboratory tests were used at Technology Test Bed.

Table 2 summarizes the results of the Technology Test Bed tests. The sensor validation system correctly tracked engine start, stop, and power level transitions, performed bias training, and monitored nominal data without issuing any false alarms. However, none of the monitored sensors experienced failures during the test series, so the sensitivity of the system in the Technology Test Bed environment could not be established. The redline sensor which caused the shutdown of Test TTB-033 was not part of the validation network being evaluated.

VI. Scaling Up

In 1992 Aerojet, under contract to LeRC, undertook the task of determining the viability of using analytical redundancy to validate the majority of the sensors used on SSME for control and health monitoring [6]. The basic approach was to identify and investigate sets of engine parameters whose measurements are statistically correlated for a nominal engine firing, or whose measurements are known to be related via first-principle (characteristic) equations.

Sensor Selection

A set of 88 engine sensors measuring 41 parameters was investigated in this task. Some of these parameters were selected as being critical to safely operating the engine, including control and redline parameters and those identified for use in advanced safety algorithms. Less critical sensors that might provide additional analytical redundancy coverage were also included. In general, only one measurement per parameter was analyzed in this task. However, several of the "redundant" sensors for certain parameters turned out to provide significantly different measurements (e.g., HPFT DS T). In these cases all redundant measurements were analyzed, resulting in a final analysis data set of 53 measurements.

The primary objective of this task was to consider the relationships between various engine parameters, thus redundant sensors were typically not evaluated. Several redundant measurements were included after an initial analytical and statistical survey identified those which showed significant differences.

Data Preparation

The data sets used consisted of nine nominal test firings for training and two additional test firings for verification. These test cases included various engines including multiple tests with the same engine serial number, thus providing useful information on test to test variations.

The sensor measurements were initially prepared by removing the start transients (first seven seconds after ignition) and the shutdown transients. The data was then smoothed and reduced from approximately one-half million data points per dataset to 50,000 data points per dataset to make the modeling procedures tractable. The reduction involved eliminating data points in time intervals during which all channels were exhibiting linear behavior (e.g., during steady-state). Only routines which computed model coefficients were run on this reduced data; all other routines, including all validation tests, used the original full sample data.

Empirical Model Selection

Initially, first and third degree binary curve fits were computed between all pairs of selected sensors. The curve fits were ranked for each test according to minimal residual variance and the rankings were averaged across the nine training test firings (e.g., if a model had the third lowest residual variation in half of the tests and the fourth lowest residual variation in the other half, its final ranking would be 3.5).

These rankings were analyzed and the top three candidate models were selected for each parameter. Other than removing redundancies from consideration, parameters were selected on the basis of their ranking and knowledge of nominal SSME operation.

The linear and cubic fit coefficients and residual characteristics for the three selected empirical relations for each sensor were then computed. Nine sets of coefficients were computed for each relation by performing linear

Test	Date	Duration (seconds)	Notes
TTB-031	4/15/92	85	Nominal firing. No false alarms, no missed detections.
TTB-032	4/28/92	205	Nominal firing. No false alarms, no missed detections.
TTB-033	5/14/92	18	Ambient powerhead temperature redline cutoff. No false alarms, no missed detections.
TTB-034	5/28/92	210	Nominal firing. No false alarms, no missed detections.
TTB-035	6/11/92	200	Nominal firing. No false alarms, no missed detections.

Table 2. Technology Test Bed Test Results

regression on each of the nine training datasets individually. A composite model was then formed for each relation by averaging the coefficients obtained for each training dataset. This composite model was then evaluated against each of the training datasets, and the mean and standard deviation of the residual computed. Finally, the average of these means and standard deviations for the composite model was computed as a measure of the overall quality of the model.

Results of Empirical Modeling

Relations were successfully developed for 33 of the 53 measurements analyzed using linear and cubic binary models. Of the remaining 20 measurements, two were found to be anomalous and six appeared amenable to multi-parameter regression modeling (i.e., appeared to be a function of more than any one other parameter). The remaining 12 measurements essentially did not correlate well to any other measurements. A good example of this set is sensor 1951 (MCC LINER CAVITY PRESSURE). According to SSME data analysts, this measures the pressure in a cavity between the inside of the MCC and the outer wall of the combustion chamber. The normal behavior for this sensor is to drop during start (as the MCC heats up, the cavity pulls a vacuum and the pressure drops) and then level off for the rest of the test. This parameter's value is thus primarily a function of time from START. These measurements which do not correlate well to other parameters can be dropped from the list of sensors evaluated by the SDV system, unless they are needed for control, redline, or health monitoring purposes, in which case a more focused modeling effort will need to be undertaken.

In summary, a large percentage of the sensors on SSME can be successfully modeled using linear and cubic polynomial regression techniques. The remaining sensors could be modeled using multi-parameter models, or other forms of models such as time-based models of nominal behavior (e.g., a function of time since engine start), or more advanced models such as neural networks. For parameters which are relatively constant during a nominal engine firing and which exist primarily to detect specific failure modes (e.g., sensor 1951 exists primarily to detect MCC burn-through), models may be developed by using data from engine anomalies or failure simulations.

Characteristic Equation Selection.

Characteristic equations are models whose forms are guided by first principles knowledge; the model coefficients are still computed empirically. The characteristic relations were identified through consideration of available sensors and knowledge of engine first principles. Of those considered, the sparsity of sensors on the SSME allowed only three types to be applied: pump flow to impeller speed, pressure rise across a pump to the square of its speed; and line resistance (pressure drop to the flow squared).

Results of Characteristic Modeling

First-principle models are difficult to derive for the SSME due to the scarcity of sensors relative to the complexity of the engine. In taking a very conservative approach, only 7 characteristic equations (with 30 parameter variations) could be fully justified as physically sound. Only one of these 7 equations failed to provide any useful predictive models (relating pressure drop across LPFP to the square of LPFP SPEED).

VII. On-Going and Future Work

This section briefly describes the technical goals of the 1993-95 LeRC Real-Time Sensor Data Validation task, and the design extensions to the AREC-based software required to meet them [7].

The primary goal of this task is to scale up the eight-channel sensor validation system developed and tested on the AREC to validate approximately 100 channels of data from the SSME in real-time. Fundamentally, this will not require any changes to the run-time algorithms, or to the established methodology for developing and validating analytical redundancy models or for tuning and validating the network.

Operation During Power-Level Transients

The AREC-based SDV system did not operate during engine start, shutdown, or power level transitions. The new SDV system will operate, at a minimum, on the latter of these transients. Thus, the system will be in continuous operation from approximately four seconds after mainstage has been achieved until the shutdown signal is detected.

To accomplish monitoring through mainstage power level transitions, the SDV system will utilize multiple sets of model thresholds and hard failure excursion thresholds. One set of thresholds will be used during steady-state (as in the AREC-hosted system), while other thresholds will be used during engine transients and during the bias training period at the start of each steady-state interval. These transient thresholds will undoubtedly be looser constraints on engine behavior. While the SDV system will not be as sensitive to soft sensor failures during the transient intervals, it will still be able to catch hard failures following two samples of anomalous data.

Integration of Advanced Models

Several groups are developing advanced models, such as neural networks, which relate data from SSME sensors [8–11]. Since these models appear to perform very well, even during power transients, they are excellent candidates for integration into the validation network as long as their run-time execution is not too computationally intensive. As NASA provides these models, Aerojet will test and integrate them into the system where needed to provide a more robust validation capability.

Software Development and Test

The SDV system developed under this task will be ported to run on a 486 and delivered to the MSFC Technology Test Bed, where it will receive data in real-time from either live engine firings, or from playbacks of previous tests. Aerojet and LeRC will be able to run the system and assess its performance remotely via network connections. The system will have a text-based display and various diagnostic routines to enable its performance to be thoroughly characterized.

VIII. Conclusion

The fundamental idea of using analytical redundancy to perform sensor data validation in real time on the Space Shuttle Main Engine has been demonstrated by the work performed on this project. However, while obtaining models for the majority of sensors used for control, redline, and advanced anomaly detection purposes appears to be straightforward, a small set of sensors may require extra modeling work if they are to be kept in the validation network.

The current work in progress will culminate in a test of the scalability of the approach, by validating data from over 100 sensors in real-time during Technology Test Bed firings. However, the ultimate goal of this work is to integrate the sensor data validation algorithms into a future enhanced Space Shuttle Main Engine controller; overall system reliability will be improved through the validation of control, redline and anomaly detection parameters during flight.

Acknowledgement

This work was supported by the NASA OACT ETO program through a NASA / Lewis Research Center Contract NAS 3-25883.

References

1. Makel, Darby, Timothy Bickmore, and William Flaspohler, *Development of Life Prediction Capabilities for Liquid Propulsion Rocket Engines — Task 3, Sensor Data Validation and Reconstruction, Phase 1: System Architecture Study Final Report*, Aerojet Propulsion Division, Contract NAS 3-25883, March 1991.
2. Pearl, J., *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, San Mateo, California, 1988.
3. Neapolitan, R., *Probabilistic Reasoning in Expert Systems: Theory and Algorithms,* Wiley Interscience, New York, 1990.
4. De Bruyne, Frank, *Probabilistic Sensor Validation*, Final Project ME290M, Department of Mechanical Engineering, University of California at Berkeley, Fall 1991.
5. *Results from Simulation Laboratory Testing of a Real-Time Sensor Data Validation System,* Aerojet Propulsion Division, 27 March 1992.
6. Bickmore, Timothy, *Real-Time Sensor Data Validation*, NASA CR-195295, March, 1994.
7. Bickmore, Timothy, "A Probabilistic Approach to Sensor Data Validation", Paper AIAA 92-3163, AIAA 28th Joint Propulsion Conference, Nashville, Tennessee, 1992.
8. Meyer, Claudia M., and William A. Maul, *The Application of Neural Networks to the SSME Startup Transient,* Paper #91-2530, 27th Joint Propulsion Conference, June, 1991.
9. Naassan, Kathryn, *Sensor Validation for the Space Shuttle Main Engine Controller,* Master's Thesis, Department of Mechanical Engineering, University of California at Berkeley, December, 1991.
10. Wheeler, Kevin, and Atam Dhawan, *Radial Basis Function Neural Networks Applied to NASA SSME Data*, Technical Report TR154/6/93/ECE, Department of Electrical and Computer Engineering, University of Cincinnati, 1993.
11. Doniere, Timothy, and Atam Dhawan, *LVQ and Backpropagation Neural Networks Applied to NASA SSME Data*, Technical Report RT156/6/93/ECE, Department of Electrical and Computer Engineering, University of Cincinnati, 1993.